Link prediction

Reference:

Guns Raf.- *Link prediction*

Measuring scholarly impact : methods and practice / Ding, Ying [edit.]; e.a. - ISBN 978-3319103761 - Berlin, Springer, 2014, p. 35-56

Link prediction

Raf Guns[1]

[1] *raf.guns@uantwerpen.be*

University of Antwerp, Institute for Education and Information Sciences, IBW,
Venusstraat 35, B-2000 Antwerpen, Belgium

Abstract

Social and information networks evolve according to certain regularities. Hence, given a network structure, some potential links are more likely to occur than others. This leads to the question of link prediction: how can one predict which links will occur in a future snapshot of the network and/or which links are missing from an incomplete network?

This chapter provides a practical overview of link prediction. We present a general overview of the link prediction process and discuss its importance to applications like recommendation and anomaly detection, as well as its significance to theoretical issues. We then discuss the different steps to be taken when performing a link prediction process, including preprocessing, predictor choice, and evaluation. This is illustrated on a small-scale case study of researcher collaboration, using the freely available linkpred tool.

Keywords: networks; network analysis; link prediction

1 Introduction

The field of informetrics studies quantitative aspects of knowledge and information. This involves areas such as citation analysis, collaboration studies, web link studies, and bibliometric mapping. These areas often comprise relations between people, documents, social structures, and cognitive structures. Although such relations can be looked at from many different angles, one of the most promising approaches is the network perspective.

The study of networks is gaining increasing attention in fields of research as diverse as physics, biology, computer science, and sociology. In informetrics as well, network analysis is a central component. This is not a new phenomenon. It was for instance recognized early on that the mathematical study of graphs might also be beneficial to a better understanding of how documents influence each other (as reflected in citation relations). Important early landmarks include work by Pinski and Narin (1976), Price (1965), and Xhignesse and Osgood (1967). Later studies have expanded on these seminal works and broadened the scope to all kinds of informational phenomena. Indeed, many interactions studied in informetrics can be represented as networks, e.g. citation networks, collaboration networks, web link networks, co-citation networks, etc. In recent years, several studies employ measures and techniques borrowed from **social network analysis** (e.g., Otte & Rousseau, 2002).

While a lengthy overview of social network analysis would lead us too far, we briefly introduce the terminology used throughout this chapter. A **network** or **graph** $G = (V, E)$ consists of a set of **nodes** or **vertices** V and a set of **links** or **edges** E. Each edge $e = \{u, v\}$ connects the nodes u and v ($u, v \in V$). The set of nodes $N(v)$ connected to a given node v is

called its **neighbourhood**. The number of nodes adjacent to v (the cardinality of the neighbourhood) is called its **degree** and denoted as $|N(v)|$.

Although networks are sometimes studied as if they are static, most social and information networks tend to be dynamic and subject to change. In a journal citation network, for instance, new journals emerge and old ones disappear, a journal may start citing a journal it had never cited before, and so on. This kind of change in a network is not entirely random; several mechanisms have been proposed that explain how networks evolve. We point out two important ones:

- **Assortativity** is the tendency of actors to connect to actors that are, in some way, similar to themselves ('birds of a feather flock together'). The criterion for similarity may be diverse: race, sex, age, interests, but also for instance node degree. In some networks **dissortativity** has been determined: the tendency to connect to others who are different from oneself ('opposites attract').
- **Preferential attachment** (Barabási & Albert, 1999) is the tendency to connect with successful actors, where success is usually measured by degree. Preferential attachment is a self-reinforcing 'rich-get-richer' mechanism, since every node that links to a high-degree node increases the latter's degree and thus its attractiveness to other nodes. This eventually evolves into a network where the degree distribution follows a power law (as is observed for many social networks). The mechanism is closely related to the Matthew effect (Merton, 1968) and Price's (1976) success-breeds-success principle.

Suppose that we have a snapshot of a network at some point in time G_t. Given the existence of mechanisms like assortativity and preferential attachment, some changes in G_t are more likely than others. For instance, the chance that two high-degree actors with similar social background will connect in the next snapshot G_{t+1} is probably higher than for two low-degree actors from different backgrounds.

Link prediction is a more formalized way of studying and evaluating this kind of intuitions. The link prediction problem can be formulated as follows (Liben-Nowell & Kleinberg, 2007): to what extent can one predict which links will occur in a network based on older or partial network data? We can distinguish between future link prediction and missing link prediction. **Future link prediction** (Guns, 2009, 2011; Guns & Rousseau, 2014; Huang et al., 2005; Spertus et al., 2005; Yan & Guns, 2014) involves predicting links in a future snapshot of the network based on a current one. **Missing link prediction** (Clauset, Moore, & Newman, 2008; Guimerà & Sales-Pardo, 2009; Kashima & Abe, 2006; Scripps et al., 2008; Zhou et al., 2009) involves predicting all links based on an incomplete or damaged version of a network (missing certain links or containing spurious ones, e.g. because of sampling or measurement errors). Both types can be addressed using similar methods.

The remainder of this chapter is structured as follows. Paragraph 2 reviews the link prediction process and its applications. In paragraph 3 we describe the data set that will be used as an example throughout the chapter. Paragraph 4 introduces the linkpred tool and paragraph 5 shows how it can be used for link prediction. Finally, the last paragraph provides the conclusions.

2 The link prediction process and its applications

Figure 1 provides a schematic overview of the link prediction process. There are four major steps: data gathering, preprocessing, prediction, and evaluation. In some studies (e.g., Guns, 2009), an extra postprocessing step in between prediction and evaluation can be distinguished. Because postprocessing is rare, it is left out here.

Data gathering may seem like an obvious step. It is explicitly included, because the quality of input data has a profound effect on the quality of later predictions. In this step, we also make a distinction between the training and the test data, both of which are typically derived from the same data sources. We will refer to the network that predictions are based on as the **training network**. If one wants to compare the prediction results to a 'ground truth' network (e.g., a later snapshot of the same network) , the latter network is called the **test network**. Note that the test network is only needed for evaluation purposes.

Preprocessing is very common but not mandatory. The main preprocessing step consists of filtering out certain nodes; the reasons and criteria for this are reviewed in paragraph 5.1. Preprocessing applies to both the training and test networks.

The prediction step operates on the training network. This step involves the choice of a **predictor**, a function or algorithm that calculates a likelihood score for each node pair (or for a subset of node pairs). Applying a predictor to the preprocessed training network yields a number of predictions. In practice, the prediction step results in a list of potential links with an associated likelihood score W. By ranking the potential links in decreasing order of W and choosing a threshold, one can obtain a predicted network. This step does not affect the training and test networks, which is reflected by the dotted lines in Figure 1.

Evaluation involves comparing the test network with the predictions. There are several possible techniques and methods for this, most of which stem from information retrieval, data mining and related field (see paragraph 5.3).

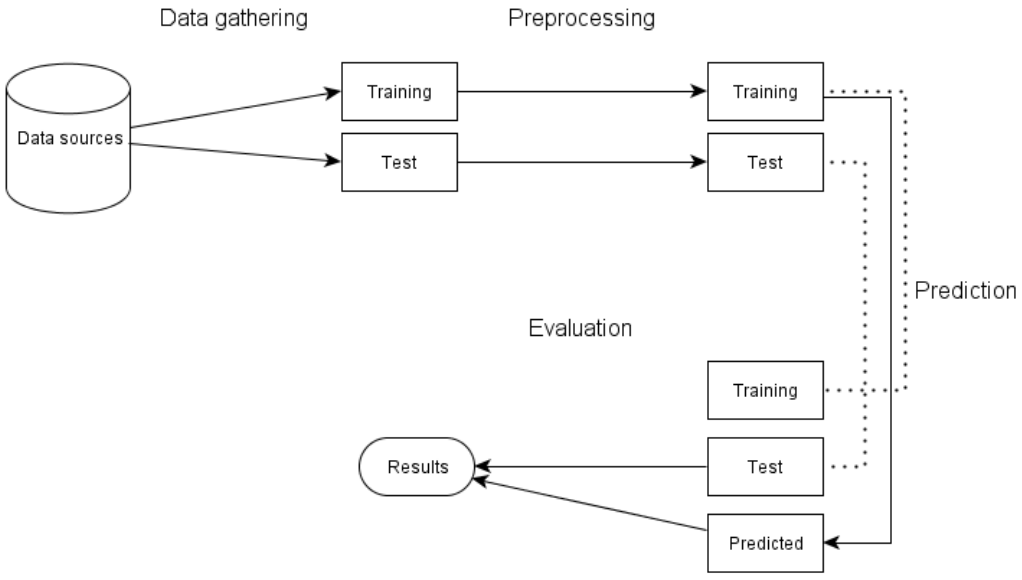

Figure 1. Overview of the link prediction process

At a high level, link prediction can be considered a statistical classification task. The items to be classified are node pairs, which should be classified into two groups: links and non-links. On the basis of empirical data (a training network, whose links we will refer to as attested links) one assigns a probability of linkage to each pair. These probability scores can then be used in different ways. We can generally distinguish between link prediction proper – that is, usage of probabilities for the actual purpose of prediction – and other applications, where these probabilities are exploited in different ways. Below, we list five possible applications of link prediction. The first two are mostly based on future link prediction, the next two build on missing link prediction, and the final one involves the importance of link prediction at a more theoretical level.

Let us first focus on link prediction proper. Typically, this happens when one is genuinely interested which links may arise in the future (or in a network which is related in a non-temporal way). In a scientometric context, this is interesting for policy makers, for whom a good understanding of likely future evolutions is crucial. Generally speaking, this use case is best served by a fairly small amount of very likely interactions or, in other words, by a focus on precision rather than recall. In the context of metadata generation, link prediction may help to alleviate sparsity of available associations (Rodriguez, Bollen, & Van de Sompel, 2009, p. 11).

A related application is **recommendation**: rather than actually trying to predict the future state of the network, one seeks to find likely but unattested links, often involving a specific node. For instance, given node a, one can create a (usually short) ranked list of candidate neighbours. These candidate neighbours are presented to a as recommendations. Interestingly, by doing so, one may influence the network's evolution. Recommendation is, for example, of great interest to smaller research groups that want to look into national or international collaboration: who are likely partners? There exists a rather large amount of literature on recommendation, recommender systems and collaborative filtering, where a and the items recommended to a are of different kinds, such as a researcher and papers she should read or cite, a library user and materials that might be of interest to him etc. Although most research into this kind of recommendation does not explicitly involve networks, it can be conceptualised as a link prediction problem within a two-mode network. Some studies explicitly study the role of link prediction in recommendation (Guns & Rousseau, 2014; Yan & Guns, 2014).

Thirdly, many networks that derive from actual data are in some ways incomplete. Link prediction can then be considered a tool to detect missing information. Conversely, erroneous data may sneak into a network and perhaps greatly affect research results. If the erroneous data cases the existence of a link that should not be present (similar to a completely random link inserted in the network), link prediction methods may single out the spurious link as highly unlikely.

Fourthly, detecting spurious links is related to a secondary application of link prediction: **anomaly detection**. Rattigan and Jensen (2005) suggest that anomaly detection offers a more fruitful line of inquiry than link prediction proper. The basic idea is that link prediction offers the tools to discover 'anomalous links', links that are unexpected and therefore interesting. For instance, an unexpected citation in a paper citation network may be a sign of interdisciplinarity (a paper building on methods or insights from other disciplines).

Finally, the main promise of link prediction at the theoretical level is as a practical way of testing and evaluating network formation and evolution models. Predictors normally derive from an explicit or implicit hypothesis of how and why links arise in a network. The performance of a predictor therefore also may help to test the validity of the underlying hypothesis. If a predictor performs markedly differently on different networks, this may point to variations in the factors that play a role in the evolution of these networks.

3 Data

We will use collaboration data to illustrate the process of link prediction. The data can be downloaded from https://raw.github.com/rafguns/linkpred/stable/examples/inf1990-2004.net and https://raw.github.com/rafguns/linkpred/stable/examples/inf2005-2009.net. The data represent a collaboration network between informetrics researchers, based on co-authorships. Thus, each node represents a researcher and each link represents a collaboration. A link's weight is the number of co-authorships of the two researchers. All data were downloaded from Thomson Reuters' Web of Science.

The training network is in the file `inf1990-2004.net` (period 1990–2004) and the test network is in the file `inf2005-2009.net` (period 2005–2009). This data set is a subset of the data used by Guns, Liu, & Mahbuba (2009) and Guns (2011, 2012). Both files are in the Pajek format and can be read and visualized with several software packages, including Pajek and VOSviewer.

Table 1 summarizes basic descriptive statistics of these networks. It can be seen that the networks are very sparse and not well connected. Although the largest component is significantly larger than the second largest component, it is not a real 'giant component' in either case. These properties may also affect the quality and feasibility of link prediction.

Table 1. Descriptive statistics of example data

	1990–2004	2005–2009
Number of nodes	632	634
Number of links	994	1052
Density	0.0050	0.0052
Number of components	95	105
Size of largest component	238	173

4 The linkpred tool

Linkpred is a tool that aims to make the most common link prediction actions available through a simple command-line interface. This means that linkpred is used by typing commands, rather than mouse input in a graphical user interface. Throughout the rest of this chapter, we assume that the operating system is Microsoft Windows. Because linkpred is written in the ubiquitous Python language (www.python.org), it can also be run under Apple's OS X and different flavours of Unix, including Linux.

An alternative software package that can be used for link prediction is LPmade (Lichtenwalter & Chawla, 2011), available from https://github.com/rlichtenwalter/LPmade.

4.1 Installation

The most straightforward way to get started is by installing the Anaconda Python distribution (https://store.continuum.io/cshop/anaconda/), which includes all required packages by default. One can then download and install linkpred as follows. Open a command line window and issue the following command:

```
> pip install https://github.com/rafguns/linkpred/archive/stable.zip
```

This will display some output as linkpred is downloaded and installed. If no error messages are shown, linkpred is successfully installed. This can be verified by trying to run it, which should display the following output:

```
> linkpred
usage: linkpred training-file [test-file] [options]
linkpred: error: too few arguments
```

If an error message states that the command is unknown, linkpred can be run by referring to its exact location, for instance:

```
> C:\Anaconda\Scripts\linkpred
```

4.2 Basic usage

Linkpred is a command-line tool. The command

```
> linkpred --help
```

should display basic usage information starting with the following line:

```
usage: linkpred training-file [test-file] [options]
```

We can see that linkpred expects a number of arguments: a training file, an optional test file, which are followed by options. The most important options are `-p` or `--predictors`, where a list of predictors is given, and `-o` or `--output`, where one can specify what kind of output is desired. A list of possible values is given in the output of `linkpred --help`.

The following examples give an idea how the tool can be used. This assumes that the current directory is the one where the data files `inf1990-2004.net` and `inf2005-2009.net` are located.

- To write all predictions based on the common neighbours predictor to a file:

```
> linkpred inf1990-2004.net -p CommonNeighbours –o cache-predictions
```

The resulting file can be found alongside the training network and will have a name of the format `<training>-<predictor>-predictions_<timestamp>.txt` (e.g., `inf1990-2004-CommonNeighbours-predictions_2013-10-25_14.47.txt`).

- To apply the common neighbours and cosine predictors, and evaluate their performance with a recall-precision chart and a ROC chart (see paragraph 5.3 for details):

```
> linkpred inf1990-2004.net inf2005-2009.net -p CommonNeighbours Cosine –o
recall-precision roc
```

The resulting charts can be found alongside the training network and will have a name of the format `<training>-<chart-type>_<timestamp>.pdf` (e.g., `inf1990-2004-ROC_2013-10-25_14.47.pdf`). If a different file format than PDF is desired, this can be obtained by setting the `-f` (or `--chart-filetype`) option to another value (e.g., `png` or `eps`).

5 Link prediction in practice

In this paragraph, we discuss the practical steps to be taken when performing a link prediction study. We discuss preprocessing, the choice of predictor(s), the actual prediction, and evaluation, all with the linkpred program. Basic information on how to use linkpred as a module within Python is provided in Appendix A.

5.1 Preprocessing

Typically, some basic preprocessing operations need to be carried out before the actual prediction takes place.

Link prediction, by definition, is the act of predicting links, not nodes. However, as networks evolve, their node set evolves as well. Hence, one cannot assume that the training and the test network both contain exactly the same set of nodes. To allow for a fair comparison, we need to restrict our analysis to nodes that are common to the training and test networks. This is automatically done by linkpred. This way, it becomes possible – at least in theory – to do a perfect prediction with 100% precision and 100% recall.

It is notoriously difficult to predict anything about isolate nodes (nodes without any neighbours), since one has no (network-based) information about them. Similarly, low-degree nodes are sometimes discarded because one has too little information about them (e.g., Liben-Nowell & Kleinberg, 2007). However, doing so may lead to overestimating the precision of the predictions (Guns, 2012; Scripps et al., 2009). By default, linkpred removes isolate nodes and leaves all other nodes intact. This can be changed by setting `min_degree` to a different value than 1 (in a profile, see paragraph 5.5).

Collaboration networks are undirected, i.e. links can be traversed in both directions. Directed networks, on the other hand, have links with an inherent direction; citation networks are a prime example. Almost all link prediction studies deal exclusively with undirected networks. Hence, linkpred has been mainly tested on undirected networks and does not support directed networks. Shibata et al. (2012) study link prediction in an article citation network. However, they essentially treat the network as undirected and only predict the presence or absence of a link between two nodes. It is then assumed that the more recent article cites the older one. A similar approach is possible with linkpred: first, convert the directed network to an undirected one, and subsequently apply link prediction.

5.2 Predictor choice

At the moment, linkpred implements 18 predictors, several of which have one or more parameters. Here, we will limit ourselves to discussing some of the most important ones. We distinguish between local and global predictors. Most predictors – both local and global – implemented in linkpred have both a weighted and an unweighted variant.

Local predictors are solely based on the neighbourhoods of the two nodes. Many networks have a natural tendency towards triadic closure: if two links $a - b$ and $b - c$ exist, there is a

tendency to form the closure $a - c$. This property is closely related to assortativity and was empirically confirmed in collaboration networks by Newman (2001), who showed that the probability of two researchers collaborating increases with the number of co-authors they have in common: "A pair of scientists who have five mutual previous collaborators, for instance, are about twice as likely to collaborate as a pair with only two, and about 200 times as likely as a pair with none." It can be operationalized by the **common neighbours** predictor (Liben-Nowell & Kleinberg, 2007):

$$W(u,v) = |N(u) \cap N(v)| \quad (1)$$

Despite its simplicity common neighbours has been shown to perform quite well on a variety of social networks: triadic closure turns out to be a powerful mechanism indeed.

The common neighbours predictor is sensitive to the size of the neighbourhood. If both u and v have many neighbours, they are automatically more likely to have more neighbours in common. Therefore, several normalizations have been introduced in the literature, such as Dice's index (Zhou, Lü, & Zhang, 2009) or the **cosine** measure (Spertus, Sahami, & Buyukkokten, 2005). The latter, for instance, is defined as:

$$W(u,v) = \frac{|N(u) \cap N(v)|}{\sqrt{|N(u)| \cdot |N(v)|}} \quad (2)$$

Other implemented normalizations include Jaccard, maximum and minimum overlap, N measure, and association strength. These indicators are fairly well-known in the information science literature, since they are frequently used as similarity measures in information retrieval and mapping studies (e.g., Ahlgren, Jarneving, & Rousseau, 2003; Boyce, Meadow, & Kraft, 1994; Salton & McGill, 1983; Van Eck & Waltman, 2009). In most empirical studies, normalizations of common neighbours have an adverse effect on performance. In other words, two nodes that have many neighbours in common are automatically likely to link to each other, regardless of their total number of neighbours. One exception to this general rule are bipartite networks, such as author–paper networks; Guns (2011) shows that the cosine measure in particular forms a good predictor for this kind of networks.

The **Adamic/Adar** predictor starts from the hypothesis that a 'rare' (i.e., low-degree) neighbour is more likely to indicate a social connection than a high-degree one. Its original definition stems from Adamic and Adar (2003) and was adapted for link prediction purposes by Liben-Nowell and Kleinberg (2007):

$$W(u,v) = \sum_{z \in N(u) \cap N(v)} \frac{1}{\log |N(z)|} \quad (3)$$

A very similar predictor is **resource allocation**, which was introduced by Zhou et al. (2009):

$$W(u,v) = \sum_{z \in N(u) \cap N(v)} \frac{1}{|N(z)|} \quad (4)$$

Resource allocation is based on a hypothesis similar to that of Adamic/Adar, but yields a slightly different ranking. In practice this predictor often outperforms Adamic/Adar. Both Adamic/Adar and resource allocation tend to yield strong predictions and outperform common neighbours.

If preferential attachment is the mechanism underlying network evolution, it can be shown that the product of the degrees of nodes u and v is proportional to the probability of a link between u and v (Barabási et al., 2002). Hence, we define the **degree product** predictor (also known as the preferential attachment predictor):

$$W(u,v) = |N(u)| \cdot |N(v)| \tag{5}$$

Note that the assumptions underlying (5) are almost the opposite of those underlying (2) and similar normalizations: whereas the former rewards high degrees, the latter punishes them. This strongly suggests that if degree product performs well as a predictor, normalized forms of common neighbours will have poor results, and vice versa. Whereas some studies report poor performance for degree product (e.g., Liben-Nowell & Kleinberg, 2007), others have found degree product to be fairly strong predictor (e.g., Yan & Guns, 2014). The cohesion and density of the network appear to be influential factors.

Many networks have weighted links. In our case study, for instance, the link of a weight is equal to the number of co-authored papers. Taking link weights into account for link prediction seems like a logical step. It is therefore quite surprising that this subject has usually been ignored (see Murata & Moriyasu, 2007 and Lü & Zhou, 2010 for some exceptions). Guns (2012) argues that the vector interpretation of set-based similarity measures (Egghe & Michel, 2002) offers a strong theoretical basis for weighted neighbour-based predictors. For instance, the weighted variant of the cosine predictor becomes:

$$W(u,v) = \frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2 \sum y_i^2}} \tag{6}$$

For each node i there is a corresponding vector element x_i or y_i. If node i is connected to u, then $x_i = w_{u,i}$, the weight of the link between u and i (likewise for v and y_i). If node i is unconnected to u (or v), the corresponding vector element is zero. This way, linkpred implements weighted versions of all local predictors.

Now let us turn to the global predictors. Even if two nodes do not share any common neighbours, they still may be related and form a link in a later stadium. A straightforward measure of relatedness is the **graph distance** between two nodes:

$$W(u,v) = \frac{1}{d(u,v)} \tag{7}$$

where $d(u,v)$ denotes the length of the shortest path from node u to v. If link weights are taken into account, graph distance is defined as:

$$W(u,v) = \left(\sum_{i=1}^{t} \frac{1}{w_i} \right)^{-1} \tag{8}$$

where w_i $(i = 1, \ldots, t)$ denotes the weight of the ith link in the shortest path (with length t) between u and v. Weighted graph distance is a much better predictor than unweighted graph distance (Guns, 2012).

In 1953, Leo Katz proposed a centrality indicator that aims to overcome the limitations of plain degree centrality (Katz, 1953). Let **A** denote the (full) adjacency matrix of the network.

The element a_{ij} is 1 if there is a link between nodes v_i and v_j or 0 if no link is present. Each element $a_{ij}^{(k)}$ of A^k (the k-th power of **A**) has a value equal to the number of walks with length k from v_i to v_j (Wasserman & Faust, 1994, p. 159). The **Katz** predictor is then defined as:

$$W(v_i, v_j) = \sum_{k=1}^{\infty} \beta^k a_{ij}^{(k)} \tag{9}$$

Generally, longer walks indicate a weaker association between the start and end node. Katz (1953) therefore introduces a parameter β $(0 < \beta < 1)$, representing the "probability of effectiveness of a single link". Thus, each walk with length k has a probability of effectiveness β^k. Its underlying hypothesis is that more and shorter walks between two nodes indicate a stronger relatedness. Guns and Rousseau (2014) show that the weighted variant of the Katz predictor can best be described in the context of a multigraph, a network where one pair of nodes can be connected by multiple links. This predictor is among the most studied and best performing of all predictors implemented in linkpred.

PageRank is well-known thanks to its implementation in the Google search engine (Brin & Page, 1998). Assume the existence of a random walker (a 'random web surfer', Page et al., 1999), who starts at a random node, randomly chooses one of its neighbours and navigates to that neighbour, again randomly chooses a neighbour and so on. Moreover, at every node, there is a small chance that the walker is 'teleported' to a random other node in the network. The chance of advancing to a neighbour is α $(0 < \alpha < 1)$ and the chance of teleportation is $1 - \alpha$. One can determine the probability that the walker is at a given node. Some nodes are more important than others and will have a higher associated probability. This probability is equal to that node's PageRank. **Rooted PageRank** is a variant of PageRank where the random walker traverses the network in the same way, except that the teleportation is not randomized: the walker is always teleported back to the same root node. The associated rooted PageRank score can be interpreted as a predictor of other nodes' relatedness to the root node.

SimRank is a measure of the similarity between two nodes in a network, proposed by Jeh and Widom (2002). The SimRank thesis can be summarized as: *nodes that link to similar nodes are similar themselves*. To compute SimRank we start from the assumption that any node is maximally similar to itself: $sim(a, a) = 1$. The unweighted formula is (Antonellis, Molina, & Chang, 2008):

$$W(u, v) = \frac{c}{|N(u)| \cdot |N(v)|} \sum_{p \in N(u)} \sum_{q \in N(v)} W(p, q) \tag{10}$$

Here, c $(0 < c < 1)$ is the 'decay factor', which determines how quickly similarities decrease. It is interesting to note that SimRank's authors explicitly acknowledge its bibliometric heritage: they regard SimRank as "a generalization of co-citation where the similarity of citing documents is also considered, recursively. […] This generalization is especially beneficial for nodes with few neighbours (e.g., documents rarely cited)" (Jeh & Widom, 2002). The performance of rooted PageRank and SimRank seems to vary from study to study; even more than for other predictors it is key to carefully tune the parameters. Furthermore, several studies use their unweighted versions, whereas the weighted ones perform better in most cases.

5.3 Prediction

Once a predictor has been chosen, it can be applied to the training network. In most cases, one wants to predict *new* links only, i.e. links that are not present in the training network. The default setting in linkpred is to predict only new links. It is, however, possible and in some cases perfectly appropriate to predict both current and new links. Anomaly detection, for instance, is based on likelihood scores that are given to current links. Current links whose likelihood score is below a certain threshold (or the bottom n) are potential anomalies. Predicting both current and new links can be enabled with the `-a` (or `--all`) option. For instance, the following command saves *all* predictions (according to SimRank) for the 2005–2009 network to a file:

```
> linkpred inf2005-2009.net -a -o cache-predictions -p SimRank
```

In the resulting list the least expected links can be considered potential anomalies, co-authorships that are unlikely given the network topology. Note that this is the opposite of 'normal' link prediction: instead of the most plausible links, we seek the least plausible ones. Without attempting to interpret the results, the six least expected co-authorships according to SimRank are the following (all of which had the same likelihood score):

- Fowler, JH – Aksnes, DW
- Liu, NC – Liu, L
- Origgi, G – Laudel, G
- Prabowo, R – Alexandrov, M
- Rey-Rocha, J – Martin-Sempere, MJ
- Shin, J – Lee, W

5.4 Evaluation and interpretation

The prediction step results in a (usually large) number of predictions, each with a certain relatedness score W. It depends on one's intentions what should be done next. If no test file is supplied, the only possible output is saving the list of predictions, by setting `output` to `cache-predictions`. The list is saved as a tab-separated file with three columns: first node, second node, and relatedness score. This format can be easily imported into a spreadsheet like MS Excel or a statistical package like SPSS for further analysis.

If a test file is supplied, it becomes possible to evaluate the prediction results by comparing them with the test network. The following additional `output` values are possible:

- `cache-evaluations`: saves the evaluation data to a tab-separated file. Each line contains the following four columns: true positive number, false positive number, false negative number, and true negative number;
- `recall-precision`: yields a recall-precision chart (this is the default);
- `roc`: yields a ROC chart, which plots the false positive rate against the true positive rate (recall)
- `f-score`: yields a chart showing the evolution of the F-score (the harmonic mean of recall and precision) as more predictions are made;
- `fmax`: yields a single-number indicator of performance, namely the highest F-score value.

Let us, as an example, try to compare the performance of the following predictors: common neighbours, cosine, degree product, SimRank, and Katz. We issue the following command:

```
> linkpred inf1990-2004.net inf2005-2009.net -p CommonNeighbours Cosine
DegreeProduct SimRank Katz
```

This will apply the chosen predictors to the training network with their default settings. Since no specific output has been specified, the default output (recall–precision) is chosen. Figure 2 displays the results. This chart illustrates the difficulty of comparing predictors. For instance, while Katz starts out as the best predictor (left side), it is overtaken by common neighbours and cosine for slightly higher recall levels. Degree product and SimRank appear to be weaker predictors in comparison, but achieve higher recall levels.

The ROC chart provides another view on the same data. Figure 3 was obtained with the command:

```
> linkpred inf1990-2004.net inf2005-2009.net -p CommonNeighbours Cosine
DegreeProduct SimRank Katz -o roc
```

This chart should be interpreted as follows. If predictions were purely randomized, they would follow the diagonal. A perfect prediction would follow a line from the bottom left to the top left corner, and from there to the top right corner. In other words, the higher, the better. Although this chart is based on exactly the same data as Figure 2, it suggests a different interpretation: here, SimRank appears to be the best predictor. The main reason is that the largest part of Figure 3 corresponds to recall values above (roughly) 0.3, which three of the five predictors do not achieve. Because most applications value precision over recall, we think recall–precision charts are usually to be preferred in link prediction evaluation.

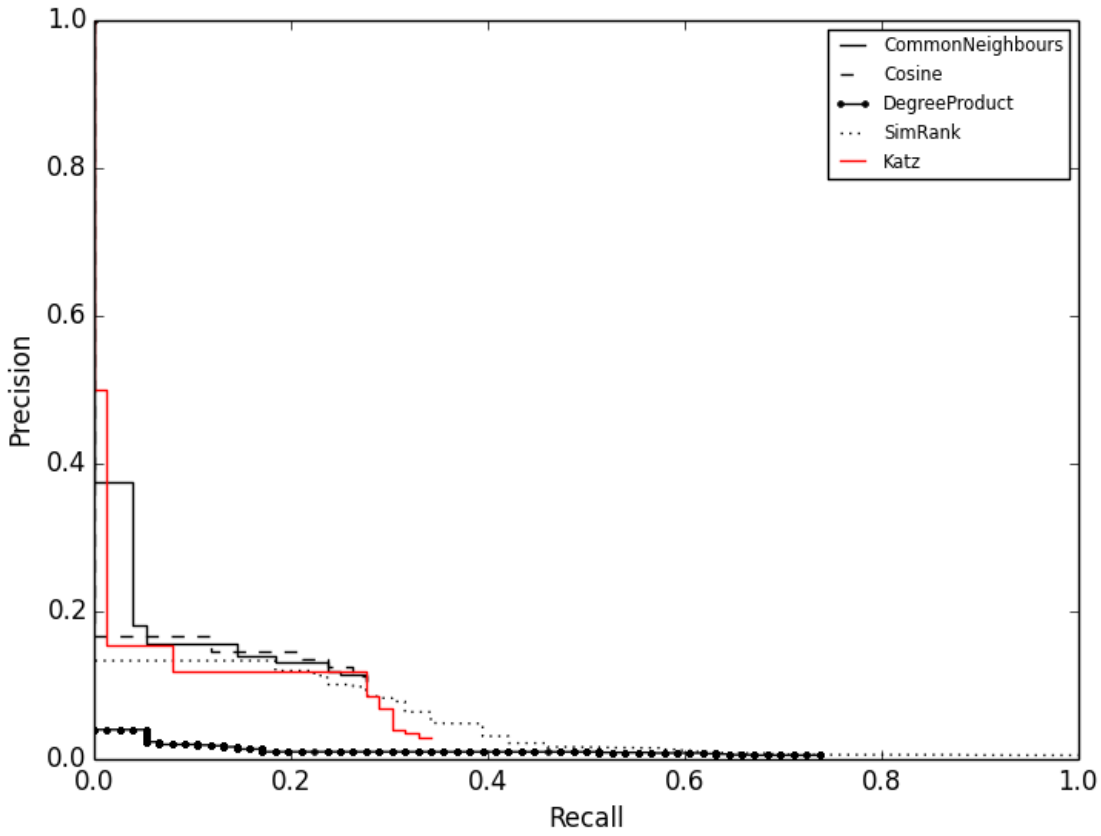

Figure 2. Recall–precision chart for five predictors

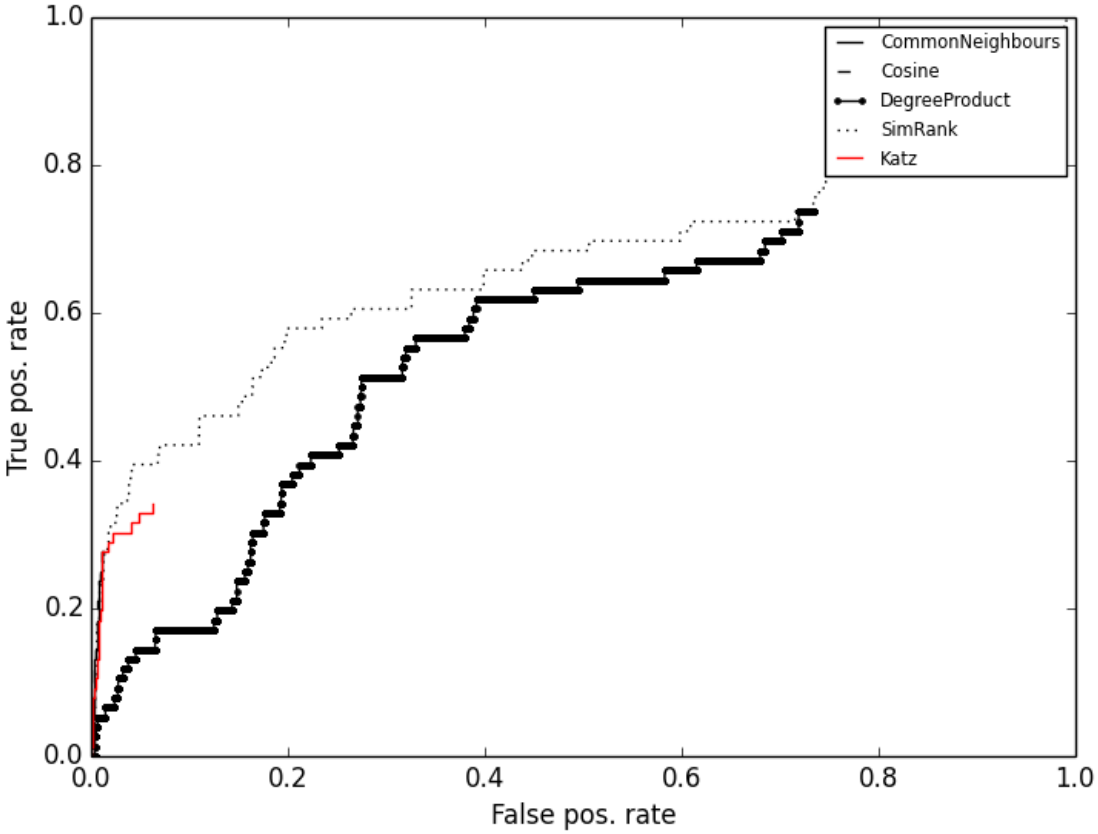

Figure 3. ROC chart for five predictors

5.5 Profiles

The linkpred tool supports the usage of **profiles**. A profile is a file that describes predictors and settings for a prediction run. Profiles have two advantages over setting predictors and settings through the command line:

1. One does not need to enter a potentially lengthy list of settings each time.
2. A profile allows for more fine-grained control over settings. For instance, predictor parameters can only be controlled through a profile.

Profiles can be written in the JSON and YAML formats.[1] We will use YAML here and demonstrate its use with a typical application: recommendation.

Suppose that we want to generate recommendations for collaboration between authors in informetrics. The main challenge here is to generate realistic recommendations that are at the same time not too obvious. As for preprocessing, it seems advisable that one does not exclude less prolific authors and sets `min_degree` to 1. The next step is prediction. Obviously we are only interested in unseen links, which is the default setting. Which predictors should be chosen? Since most local predictors yield rather obvious predictors, one will probably mainly use global predictors with good performance, such as rooted PageRank and Katz. The parameters of these predictors influence how far apart a predicted node pair can be; we choose to test two different values for each. Finally, since we are recommending, there is no (formal) evaluation step and we set `output` to `cache-predictions`. The corresponding profile looks like this:

[1] See, e.g., http://en.wikipedia.org/wiki/JSON and http://en.wikipedia.org/wiki/YAML.

```
output:
- cache-predictions
predictors:
- name: RootedPageRank
  displayname: Rooted PageRank (alpha = 0.5)
  parameters:
    alpha: 0.5
- name: RootedPageRank
  displayname: Rooted PageRank (alpha = 0.8)
  parameters:
    alpha: 0.8
- name: Katz
  displayname: Katz (beta = 0.01)
  parameters:
    beta: 0.01
- name: Katz
  displayname: Katz (beta = 0.001)
  parameters:
    beta: 0.001
```

Note that we can set predictor parameters and change a predictor's display name (the way it is displayed in chart legends etc.) We save the profile as `rootedpr.yaml` and use it as follows:

```
> linkpred inf2005-2009.net --profile rootedpr.yaml
```

Because it is hard to assess which parameter values will yield the best predictions, it is often advisable to first test different settings on a related set of training and test networks. For instance, since we are recommending new links on the basis of the 2005–2009 data set, we could test which α value for rooted PageRank yields the best result for the 1990–2004 training network and the 2005–2009 test network. As indicated by the recall–precision chart (not shown) and the maximum F-scores (Table 2), a value around 0.5 appears to be optimal.

Table 2. Maximum F-scores for rooted PageRank with different α values

α	*F*max
0.1	0.1538
0.5	0.1923
0.9	0.1554

As a last example, we show how to compare the weighted and unweighted variants of a predictor, in this case Jaccard, with a recall–precision and ROC chart. The profile looks like this:

```
output:
- recall-precision
- roc
predictors:
- name: Jaccard
  displayname: Jaccard, unweighted
  weight: null
```

```
- name: Jaccard
  displayname: Jaccard, weighted
  weight: weight
```

6 Discussion and conclusions

The use of network analysis techniques has become increasingly common in informetric studies over the last ten years. Link prediction is a fairly recent set of techniques with both theoretical and practical applications. The techniques were illustrated on a case study of collaboration between researchers in the field of informetrics.

Generally speaking, it turns out that link prediction *is* possible, because link formation in social and information networks does not happen at random. Nonetheless, link prediction methods are also limited in several ways. First, these methods rely only on network topology and are unaware of social, cognitive and other circumstantial factors that may affect a network's evolution. To some extent, such factors are reflected in the network's topology but the match is always imperfect. Second, the question which predictors are the best choice in a given concrete situation is difficult to answer. Some predictors (e.g., Katz) exhibit good performance in many studies and are therefore a good 'first choice' in the absence of more specific information. At the same time, it is typically a good idea to test different predictors on a comparable data set (e.g., Yan & Guns, 2014) in order to make a more informed decision on predictor choice. Third, there is a trade-off between prediction accuracy and non-triviality: good predictors often yield predictions that are predictable (!) and therefore less interesting.

As we have shown, the linkpred tool offers a simple but powerful way to perform link prediction studies. Its main limitations pertain to speed and network size: very large networks may be slow or even impossible to analyze with linkpred (also depending on which predictors are used). Nevertheless, the program has been applied to networks with several thousands of nodes and links without any problems. Moreover, it is always possible to export predictions (and evaluations) from linkpred for further analysis and processing in other software. Finally, we mention that linkpred is available as open source software (modified BSD license).

7 References

Adamic, L., & Adar, E. (2003). Friends and neighbors on the web. *Social Networks*, 25(3), 211–230.

Ahlgren, P., Jarneving, B., & Rousseau, R. (2003). Requirements for a cocitation similarity measure, with special reference to Pearson's correlation coefficient. *Journal of the American Society for Information Science and Technology*, 54(6), 550–560.

Antonellis, I., Molina, H.G., & Chang, C.C. (2008). Simrank++: query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment*, 1(1), 408–421.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439), 509.

Barabási, A.-L., Jeong, H., Néda, Z., Ravasz, E., Schubert, A., & Vicsek, T. (2002). Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications*, 311(3-4), 590–614.

Boyce, B.R., Meadow, C.T., & Kraft, D.H. (1994). *Measurement in Information Science*. San Diego: Academic Press.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7), 107–117.

Clauset, A., Moore, C., & Newman, M.E.J. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191), 98–101.

Egghe, L., & Michel, C. (2002). Strong similarity measures for ordered sets of documents in information retrieval. *Information Processing & Management*, 38(6), 823–848.

Guimerà, R., & Sales-Pardo, M. (2009). Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52), 22073–22078.

Guns, R. (2009). Generalizing link prediction: Collaboration at the University of Antwerp as a case study. *Proceedings of the American Society for Information Science & Technology*, 46(1), 1–15.

Guns, R. (2011). Bipartite networks for link prediction: can they improve prediction performance? In E. Noyons, P. Ngulube, & J. Leta (Eds.), *Proceedings of the ISSI 2011 Conference* (pp. 249–260). Durban: ISSI, Leiden University, University of Zululand.

Guns, R. (2012). *Missing links: Predicting interactions based on a multi-relational network structure with applications in informetrics*. Doctoral dissertation, Antwerp University.

Guns, R., Liu, Y., & Mahbuba, D. (2011). Q-measures and betweenness centrality in a collaboration network: a case study of the field of informetrics. *Scientometrics*, 87(1), 133–147.

Guns, R. & Rousseau, R. (2014). Recommending research collaborations using link prediction and random forest classifiers. *Scientometrics*, in press. http://dx.doi.org/10.1007/s11192-013-1228-9

Huang, Z., Li, X., & Chen, H. (2005). Link prediction approach to collaborative filtering. In *JCDL'05: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries* (pp. 141–142). NY: ACM Press.

Jeh, G., & Widom, J. (2002). SimRank: a measure of structural-context similarity. In *KDD '02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 538–543). New York: ACM.

Kashima, H., & Abe, N. (2006). A parameterized probabilistic model of network evolution for supervised link prediction. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM2006)* (pp. 340–349). Washington DC: IEEE Computer Society.

Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, 18(1), 39–43.

Langville, A.N., & Meyer, C.D. (2005). A survey of eigenvector methods for web information retrieval. *SIAM Review*, 47(1), 135–161.

Liben-Nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7), 1019–1031.

Lichtenwalter, R.N., & Chawla, N.V. (2011). LPmade: Link prediction made easy. *Journal of Machine Learning Research*, 12(Aug): 2489–2492.

Lü, L., & Zhou, T. (2010). Link prediction in weighted networks: The role of weak ties. *EPL (Europhysics Letters)*, 18001.

Murata, T., & Moriyasu, S. (2007). Link prediction of social networks based on weighted proximity measures. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 85–88). Washington, DC: IEEE Computer Society.

Newman, M.E. (2001). Clustering and preferential attachment in growing networks. *Physical Review E*, 64(2), 025102.

Opsahl, T., Agneessens, F., & Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3), 245–251.

Otte, E., & Rousseau, R. (2002). Social network analysis: a powerful strategy, also for the information sciences. *Journal of Information Science*, 28(6), 441–453.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank citation ranking: Bringing order to the web*. Stanford, CA: Stanford Digital Library Technologies Project.

Pinski, G. & Narin, F. (1976). Citation influence for journal aggregates of scientific publications: Theory with application to the literature of physics. *Information Processing & Management*, 12(5), 297–312.

Price, D.J. de Solla. (1965). Networks of scientific papers. *Science*, 149(3683), 510–515.

Price, D.J. de Solla. (1976). A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5), 292–306.

Rattigan, M.J., & Jensen, D. (2005). The case for anomalous link discovery. *ACM SIGKDD Explorations Newsletter*, 7(2), 41–47.

Rodriguez, M.A., Bollen, J., & Van de Sompel, H. (2009). Automatic metadata generation using associative networks. *ACM Transactions on Information Systems*, 27(2), 1–20.

Salton, G., & McGill, M.J. (1983). *Introduction to Modern Information Retrieval*. New York: McGraw-Hill.

Scripps, J., Tan, P.N., & Esfahanian, A.H. (2009). Measuring the effects of preprocessing decisions and network forces in dynamic network analysis. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 747–756). New York: ACM.

Shibata, N., Kajikawa, Y., & Sakata, I. (2012). Link prediction in citation networks. *Journal of the American Society for Information Science and Technology*, 63(1), 78–85.

Spertus, E., Sahami, M., & Buyukkokten, O. (2005). Evaluating similarity measures: a large-scale study in the Orkut social network. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (pp. 678–684). New York: ACM.

Van Eck, N.J., & Waltman, L. (2009). How to normalize cooccurrence data? An analysis of some well-known similarity measures. *Journal of the American Society for Information Science and Technology*, 60(8), 1635–1651.

Wasserman, S., & Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Cambridge: University Press.

Xhignesse, L.V., & Osgood, C.E. (1967). Bibliographical citation characteristics of the psychological journal network in 1950 and in 1960. *American Psychologist*, 22(9), 778–791.

Yan, E., & Guns, R. (2014). Predicting and recommending collaborations: An author-, institution-, and country-level analysis. *Journal of Informetrics*, 8(2), 295–309.

Yang, LY. & Jin, BH. (2006). A co-occurrence study of international universities and institutes leading to a new instrument for detecting partners for research collaboration. *ISSI Newsletter*, 2(3), 7–9.

Zhou, T., Lü, L., & Zhang, Y.-C. (2009). Predicting missing links via local information. *European Physical Journal B*, 71(4), 623–630.

Appendix A. Usage as a Python module

Linkpred can be used both as a standalone tool and as a Python module. Here we provide basic instructions for usage as a Python module.

Once linkpred is properly installed, we can open a Python console and load the module from within Python.

```
> python
Python 2.7.3 (default, Apr 10 2012, 23:24:47) [MSC v.1500 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import linkpred
```

The linkpred module is now loaded and can be used. First, let us open a network:

```
>>> G = linkpred.read_network("inf1990-2004.net")
11:49:00 - INFO - Reading file 'inf1990-2004.net'...
11:49:00 - INFO - Successfully read file.
```

We can now explore some properties of the network (stored in variable `G`), such as its number of nodes or links (see the NetworkX documentation at http://networkx.github.io/ for further information):

```
>>> len(G)   # number of nodes
632
>>> G.size() # number of links
994
```

Predictors can be found in the `linkpred.predictors` submodule. Let us create a SimRank predictor for our network as an example. By setting `only_new` to `True`, we make sure that we only predict new links (i.e., links that are not present in the current network).

```
>>> simrank = linkpred.predictors.SimRank(G, only_new=True)
```

The line above only sets up the predictor, it does not actually apply it to the network. To do that, we invoke the `predict` method. Predictor parameters can be set here; we will set *c* to 0.5.

```
>>> simrank_results = simrank.predict(c=0.5)
```

Finally we take a look at the top 5 predictions and their scores.

```
>>> top = simrank_results.top(5)
>>> for authors, score in top.items():
...     print authors, score
...
Tomizawa, H - Fujigaki, Y 0.188686630053
Shirabe, M - Hayashi, T 0.143866427916
Garfield, E - Fuseler, EA 0.148097050146
Persson, O - Larsen, IM 0.138516589957
Vanleeuwen, TN - Noyons, ECM 0.185040358711
```